클러스터 시스템 환경하에서의 병렬 CBF 기법의 구현 및 성능 평가

박승봉[0] 장재우

전북대학교 컴퓨터공학과

(sbpark[0], jwchang)@dblab.chonbuk.ac.kr

Implementaion and Performance Analysis of a Parallel CBF Scheme under Cluster System Environment

Seong-Bong Park[0] Jae-Woo Chang

Dept. of Computer Engineering, Chonbuk National University

요 약

기존의 색인 기법들은 차원의 수가 증가할수록 검색 성능이 급격히 저하되는 문제를 지니고 있으며, 이를 극복하기 위하여 CBF 기법이 제안되었다. 그러나 CBF 기법은 데이터 양이 증가함에 따라 검색성능이 선형적으로 감소하는 문제가 존재한다. 이를 해결하기 위해 다수의 디스크를 수평 분할 방법을 이용하여 디클러스터링(declustering)을 하는 병렬 CBF 기법이 제안되었다. 본 논문에서는 병렬 CBF기법을 여러 대의 리눅스 컴퓨터를 이용한 클러스터 시스템 환경하에서 구현하고, 삽입시간, 범위 질의 검색시간, k-최근접 질의 검색시간 측면에서 성능 평가를 수행한다. 아울러, 클러스터 시스템 환경하에서의 병렬 CBF 기법을 기존 CBF 기법과 성능 비교를 수행하며, 이를 통해 병렬 CBF 기법이 기존 CBF 기법보다 우수한 검색 성능을 나타냄을 보인다.

1. 서 론

데이터웨어하우징에 사용되는 데이터의 n-차원 속성이나 멀티미디어 데이터베이스에서 사용하는 멀티미디어 객체의 n-차원 특징 벡터들은 모두 고차원 데이터이며, 이러한 고차원 데이터를 효율적으로 검색하기 위해서는 고차원 색인 기법이 요구된다. 제안된 대부분의 색인 기법들이 차원의 수가 증가할수록 검색 성능이 급격히 저하되는 '차원 저주(dimensional curse)' 문제를 지니고 있다[1]. 이러한 문제를 해결하기 위해 VA-File과 CBF 기법이 제안되었다. VA-File 기법은 근사 정보를 사용하여 필터링을 수행하며[2], CBF(Cell-Based Filtering) 기법은 VA-File 기법의 근사정보 외에 셀에 적합한 최대거리 및 최소거리를 새롭게 정의하여 시그니처를 구성함으로써 우수한 검색 성능을 제공한다[3].

그러나, VA-File이나 CBF기법은 데이터의 양이 증가할수록 선형적으로 검색성능이 감소하는 문제가 존재한다. 이를 해결하기 위해 수평 분할 방법을 이용하여 데이타를 디클러스터링(declustering)하는 병렬 CBF 기법이 제안되었다[4]. 본 논문에서는 수평 분할 방법을 이용한 병렬 CBF 기법을 여러 대의 리눅스(Linux) 컴퓨터를 이용한 클러스터 시스템 환경하에서 구현한다. 아울러, 삽입시간, 범위 질의 검색시간, k-최근접 질의 검색시간 측면에서 성능 평가를 수행하며, 마지막으로 클러스터 시스템 환경하에서의 병렬 CBF 기법을 기존 CBF 기법과 성능 비교를 수행한다.

본 논문의 구성은 다음과 같다. 2장에서는 클러스터 시스템 환경하에서의 병렬 CBF 기법의 구현에 대하여 기술하고, 3장에서는 클러스터 시스템 환경하에서의 병렬 CBF 기법의 성능평가에 대하여 기술하고, 마지막으로 4장에서는 결론 및 향후 연구 방향을 기술한다.

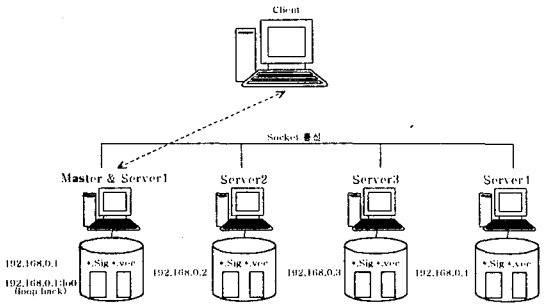
2. 클러스터 시스템 환경하에서의 병렬 CBF기법의 구현

CBF 기법은 객체들의 특징 벡터들에 대한 순차 탐색을 수행하기 전에, 각각의 특징 벡터에 대한 시그니처를 탐색하여 필터링함으로써 탐색 성능을 향상시킨다. 이러한 특징벡터를 시그니처로 변환하는데 있어서[5], 셀(Cell)로 표현된 벡터의 정보와 셀 중심간의 거리 정보를 이용하여 필터링 효과를 증대시킴으로써 검색성능을 향상시킨다. 아울러, 시그니처 파일은 전체 탐색되어야 하고, 특징 벡터 파일은 시그니처 파일의 탐색을 통해 얻어진 후보 특징 벡터만 부분적으로 탐색되어지는 특성을 고려하여, 시그니처와 특징 벡터 데이터를 수평 분할 방법[6]을 이용하여 디클러스터링하는 병렬 CBF 기법이 제안되었다[4].

한편 병렬 CBF 기법의 성능향상을 위해, 클러스터 시스템 환경하에서 시그니처와 특징 벡터 데이터를 각 클러스터 컴퓨터에 분산시켜 저장 및 검색하는 병렬 CBF 기법의 구현이 요구된다.[그림 1]은 수평 분할 방법을 이용한 병렬 CBF 기법을 여러 대의 리눅스(Linux) 컴퓨터를 이용한 클러스터 시스템 환경하에서 구현하는 아키텍처(Architecture)를 나타낸다. 이는 물리적으로 분리된 다수의 컴퓨터에 시그니처 데이터와 특징 벡터 데이터를 병렬 분산시켜 저장 및 검색을 수행한다.

이의 구현을 위하여, socket방식의 통신을 사용한다. 즉, 하나의 Master가 자신을 포함한 4대의 서버(Server)에 생성, 삽입, 질의 검색과 같은 오퍼레이션을 요구하면, 4대의 서버에서 이를 처리하고, Master에 결과를 반환하는 방식을 사용한다. 여기에 각각의 서버를 독립적으로 수행시키기 위하여 Master는 서버를 관리하는 스레드를 유지하며, 데이터 삽입시에 벡터 데이타를 각각의 서버에 수평분할 방법을 이용하여 CBF 파일을 생성하고, 질의파일을 읽어들여 서버에 전달한다. 각각의 서버는 Master로부터 질의를 받고, 이를 처리하기 위해 이미 생성된 CBF파일을 검색하여 질의결과를

Master로 반환한다. 그리고 Master는 반환된 결과를 통합하여 필요한 경우 재계산을 수행하고, 이를 통하여 검색 성능을 측정한다.



[그림 1] 클러스터 시스템 환경하에서 구현된 병렬CBF 기법의 아키텍처

한편 클러스터 시스템 환경하에서의 병렬 CBF 기법의 구현을 위하여, 리눅스 시스템이 사용하는 하드디스크와 별도로 클러스터 시스템의 서버가 사용하는 하드디스크를 마운트(Mount)하여 사용한다. 이는 리눅스 시스템에서 OS(Operating System)가 시스템을 접근하는데 있어 Swap 영역을 사용하기 때문에, 병렬 CBF의 성능에 영향을 미치는 하드디스크를 접근하는 OS의 영향을 줄이기 위함이다. 삽입시 벡터 데이타를 서버에 전달할 때와 범위질의와 K최근접 질의 처리의 결과를 Master에 반환할 때, 통신 과정의 오버헤드(Overhead)를 감소시키기 위해 8Kbyte 크기의 버퍼를 사용한다. 한편 서버 1은 루프백 어드레스(loop-back)를 이용하여 Master이자 서버의 구조를 가지며, 서버 2,3,4는 별도의 허브(Hub)에 연결하여 내부 네트워크를 구성함으로써 보다 빠른 네트워크 환경을 구성한다.

3. 성능 평가

[표 1] 실험적 성능평가 시스템 환경

OS	Red hat Linux 7.0 2.4.5
CPU	450 MHZ
disk	HDD 30GB
Main Memory	128 MB
Compiler	gcc 2.96

본 절에서는 클러스터 시스템 환경하에서 병렬 CBF 기법을 구현하여 성능 평가를 수행한다. 구현된 클러스터 시스템 환경하에서의 하나의 리눅스 컴퓨터는 [표 1]과 같다. 실험 데이터는 200만건의 Synthetic 랜덤 데이터 셋 각각 10차원, 20차원, 40차원, 50차원, 80차원,100차원을 사용한다. 성능 비교 대상은 클러스터 시스템 환경하에서 구현된 병렬 CBF 기법(이후 Cluster)과 기존 CBF 기법(이후 Single)이다. 또한 성능 평가는 데이터 삽입, 범위 질의 검색, k-최근접 질의 검색에 필요한 시간을 측정하여 수행한다.

3.1 삽입 시간

삽입 시간은 클러스터 시스템 환경하에서의 병렬 CBF 기법에 200만건의 Synthetic 랜덤 데이터를 삽입하는데 필요한 시간이며, 여기서 비율을 기존 CBF기법의 삽입시간을 병렬 CBF기법의 삽입시간으로 나눈 값이다. 삽입시간은 [표 2]와 같다.

삽입 시간

(단위:초, %)

10차원	237	256	92
20차원	479	445	108
40차원	828	824	101
50차원	993	1,002	99
60차원	1,170	1,188	98
80차원	1,506	1,544	98
100차원	1,862	1,930	96

[표 2] 삽입 시간

[표 2]에서 알 수 있듯이 10, 50, 100차원의 경우 기존 CBF 기법은 삽입시간이 236초, 993초, 1,862초가 소요되며, 병렬 CBF의 경우 256초, 1,002초, 1930초가 소요된다. 두 기법 모두 차원이 증가함에 따라 데이터 삽입 시간도 순차적으로 증가함을 알 수 있다. 또한 기존 CBF 기법과 병렬 CBF 기법은 삽입 시간 측면에서 성능이 거의 비슷함을 알 수 있다.

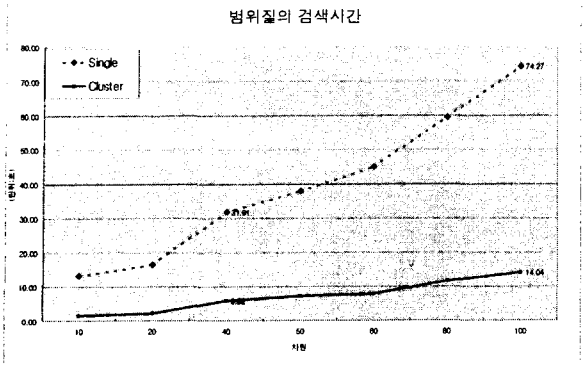
기존 CBF 기법과 병렬 CBF 기법이 삽입시간 측면에서 비슷한 성능을 보이는 이유는 다음의 이유 때문이다. 즉, CBF 파일을 생성할 때 입력 디스크로 부터 벡터 데이타를 읽어들이는 시간과 이를 새로운 파일로 쓰는 시간이 필요하다. 이때 읽는 시간은 동일하며, 쓰는 시간이 여러 대의 컴퓨터에서 동시에 수행되기 때문에 병렬 CBF 기법의 삽입시간은 줄어들 수 있다. 그러나 병렬 CBF 기법은 네트웍을 이용하여 삽입을 수행하기 때문에 통신하는 사이에서의 오버헤드가 존재한다. 이러한 이유 때문에 전체적으로 기존 CBF 기법과 병렬 CBF 기법은 삽입시간이 거의 동일하다. 한편, Master와 서버의 데이터 교환시 네트웍이 허용하는 충분한 버퍼를 할당하여 고속의 네트웍이 가능하도록 하면, 통신 오버헤드를 줄일 수 있다.

3.2 범위 질의 검색 시간

범위 질의는 주어진 질의 포인트로부터 일정한 거리 영역 안에 포함된 객체를 찾는 검색이다. 병렬 CBF 기법의 성능 평가를 위하여 물리적으로 독립된 디스크를 리눅스 시스템에서 사용한다. 성능평가를 위한 범위 값은 전체 데이터 중에서 0.1%의 데이터를 검색할 수 있는 값을 사용한다. [그림 2]는 범위 질의 검색 시간을 기존 CBF 기법(Single)과 병렬 CBF 기법(Cluster)에서 서버가 4개인 경우를 각 차원별로 비교한 것이다.

데이터의 차원수가 10차원일 때 병렬 CBF 기법은 기존 CBF 기법과 비교하여 830%정도의 성능향상을 보이며 이는 입력 데이터를 4대의 서버로 수평 분할하여 저장함에 따라 Page I/O 수가 크게 줄어들기 때문이다. 한편 40차원의 경우 539%, 100차원의 경우 529%로 성능이 향상됨을 알 수 있다. 이는 클러스터 시스템을 적용할 때의 장점으로 용량이 큰 파일을 n개로 나누었을 때, 디스크를 접근하는 Page I/O수는 1/n이하로 줄어들며, 버퍼 방식을 적용할 경우 Page I/O수가 더욱 줄어들기 때문이

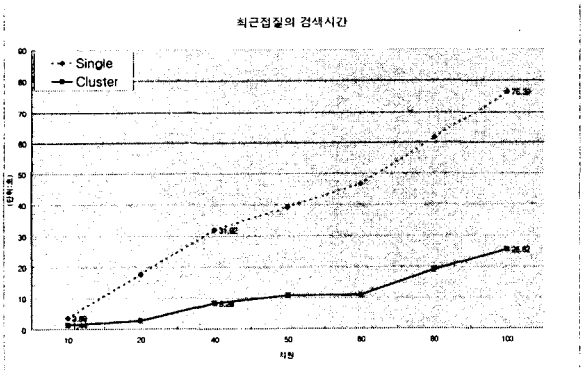
다.



[그림 2] 범위질의 검색시간

3.3 k-최근접 질의 검색 시간

k-최근접 질의는 주어진 질의 벡터로부터 가장 가까운 k개의 객체를 찾는 검색이다. 병렬 CBF 기법의 성능 평가를 위해 k값을 100으로 정하여 검색 시간을 측정한다. [그림 3]은 기존 CBF 기법과 병렬 CBF 기법의 10차원, 20차원, 40차원, 50차원, 60차원,80차원,100차원에서 측정된 k-최근접 질의 검색 시간을 나타낸다.



[그림 3] K-최근접 질의 검색시간

데이터의 차원수가 10차원일 때, 기존 CBF기법은 1.44초, 병렬CBF 기법은 3.59초로 병렬 CBF 기법이 기존 CBF 기법과 비교하여 249%정도의 성능향상을 보이며 이는 입력 데이터를 4대의 서버로 수평 분할하여 저장함에 따라 Page I/O 수가 크게 줄어들기 때문이다. 한편 40차원의 경우 기존 CBF 기법이 31.92초, 병렬CBF 기법이 8.28초로 386%의 성능향상이 있으며, 아울러 100차원인 경우 기존 CBF 기법이 76.39초, 병렬 CBF 기법이 25.62초로 298 %의 성능향상을 볼 수 있다. 이는 범위질의 검색 시간보다 성능향상은 적지만 거의 300%정도의 성능향상을 보임을 알 수 있다.

한편 범위질의와 비교할 때 성능향상이 저조한 이유는, 저장된 벡터 데이터의 분포에 따라 서버마다 k 개의 최근접 벡터 데이터를 검색하는 시간이 가변적이기 때문이다. 즉, 병렬 CBF 기법에서 k-최근접 질의를 처리함에 있어 다른 서버에 비해 상대적으로 느린 서버가 전체 성능을 좌우하기 때문이다. 아울러 k-최근접 질의를 처리함에 있어 클러스터 컴퓨터의 수가 N이라면 검색시간이 1/N으로 줄어들지 않는 이유는, k-최근접 질의의 특성상 각 서버마다 k개의 최근접 벡터를 찾고, 아울러 Master에서 이를 통합하여 최종적인 k개의 최근접 벡터를 재계산하는 시간이 부가적으로 필요하기 때문이다.

4. 결론 및 향후 연구

본 논문에서는 수평 분할 방법을 이용한 병렬 CBF 기법을 Linux 클러스터 시스템 환경하에서 구현하고, 삽입시간, 범위 질의 검색시간, k-최근접 질의 검색시간 측면에서 성능 평가를 수행하였다. 아울러, 클러스터 시스템 환경하에서 구현된 병렬 CBF 기법을 기존 CBF 기법과 성능 비교를 수행하였다. 기존 CBF 기법과 성능 비교,결과, 병렬 CBF 기법이 범위 및 k-최근접 질의에 대해서 차원에 관계없이 클러스터 컴퓨터 개수에 따라 선형적으로 성능 향상이 이루어짐을 보였다.

향후 연구로는 k-최근접 질의에 대해 검색 성능을 보다 개선할 수 있는 방법을 연구하고, 기존의 VA-File을 클러스터 시스템 환경하에서 구현하여 병렬 CBF 기법과 성능비교를 수행하는 것이다.

참고문헌

[1] S. Berchtold, C. Bohm, D. Keim, and H.-P. Kriegel, "A Cost Model for Nearest Neighbor Search in High-Dimensional Data Space", ACM PODS Symposium on Principles of Databases Systems, Tucson, Arizona, 1997.

[2] R. Weber, H.-J. Schek, S. Blott, "A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces," Proceedings of International Conference on Very Large Data Bases, pp.24-27, 1998.

[3] S.-G. Han and J.-W. Chang. "A New High-Dimensional Index Structure Using a Cell-based Filtering Technique", In Lecture Notes in Computer Science 1884 (Current Issues in Databases and Information Systems), Springer, pp. 79-92, 2000.

[4] 김남기, 장재우 " 수평 분할 방법을 이용한 병렬 CBF(Cell-Based Filtering) 기법의 설계", 한국정보과학회 추계 학술발표 논문집(I), 제 28권 2호, pp. 70-72, 2001.

[5] C. Faloutsos, "Design of a Signature File Method that Accounts for Non-Uniform Occurrence and Query Frequencies", ACM SIGMOD, 165-170, 1985.

[6] J.-K. Kim and J.-W. Chang, "A Horizontally Divided Signature File on Parallel Machine Architecture," Journal of System Architecture, Vol. 44, No. 9-10, pp.723-735, Jun. 1998.